REMARKS

Claims 1 – 12 are pending in the Application. The Examiner has rejected Applicant's declaration as being insufficient to overcome Jordan, US Patent Publication No. 2002/0073323, filed July 14, 2001 and claiming priority to Provisional Application No. 60/218,333, filed July 14, 2000 ("Jordan.") Applicant respectfully traverses the rejection.

The Examiner's rejection states, at page 2, that the evidence provided does not disclose "writing the results of said interpretation." But with all due respect, the Examiner is incorrect. The statement regarding "Build Up table of values, addresses, flags and results" is the statement that discloses writing the results of said interpretation. The writing is at least inherent in the building up, and Applicant would argues is in fact directly shown by the statement. That is, in order to build up, the program must write the results.

Applicant respectfully disagrees with the Examiner's statement that there is no disclosure of interpreting code. The interpreter is what interprets – and macros are code. Thus, the presence of a macro interpreter provides the disclosure showing that code is interpreted – and so Applicant submits the disclosure does provide evidence of conception of interpreting code.

Examiner's statement that there is no disclosure of computer readable medium is incorrect. Macros are generally contained upon computer readable medium, and are interpreted usually during a read from that media. Thus the reference to macro interpreter should be taken as a showing of computer readable media.

Finally, a result evaluator and a reporter are seen on the third page, where the manager is asked to make a decision after the virus scan reports. (If infected, the code will be denied – which shows a results evaluator.) The first page also shows a good/no good designation, which, it is submitted, provides a results evaluator as well as reporter.

The Examiner has also rejected the affidavit as failing to show due diligence between conception and reduction to practice. With all due respect, the affidavit properly and clearly recites that due diligence was exercised. Therefore it is competent evidence as to that exercise. The Examiner's rejection on the grounds that the recitation of evidence in the affidavit is insufficient is misplaced – and Applicant respectfully requests the withdrawal of that rejection.

In summary, Applicant submits that the affidavit properly provides for removal of the Jordan reference from the present case.

Applicant also notes that the Jordan reference is not prior art to the claims in any event.

Claims 1-4, and 6 are rejected under 35 U.S.C. 103(a) as being unpatentable over Jordan. Applicant respectfully traverses the rejection.

As to claim 1, Applicant respectfully but strenuously disagrees with the Examiner's statement that an emulator and interpreter are interchangeable, as neither executes code. An emulator is "designed to make one part of computer or component act as if it were another. By means of an emulator, a computer can run software written for another machine." Microsoft Computer Dictionary, 5th Ed. 2002, at p. 191 (enclosed.) An interpreter is a "program that translates and then executes each statement in a program written in an interpreted language." *Id.*, at 285 (enclosed.) Thus, neither definition is in

accord with the Examiner's statement – both specifically execute code.

Applicant has amended the claims at issue to make more clear the nature of the interpreter herein. It is an evaluative interpreter and so does not execute code. Moreover, as such it is even further away from the emulator of Jordan. That emulator executes the code provided to it. The evaluative interpreter of the claim evaluates and interprets code. Thus, Applicant submits, the executable emulator of Jordan cannot be compared to the evaluative interpreter of the present claims.

There are other patentable differences between claim 1 and Jordan. For example, Applicant is unable to find anywhere in Jordan any mention of scanning any results of any interpretation. (Of course as noted above, Jordan does not interpret – rather Jordan executes as a result of its emulation. Therefore, Jordan would not scan any results of any interpretation.)

Neither does Jordan scan any results of its emulation. Rather Jordan "monitors," as was noted by the Examiner (paragraph 28 of Jordan.) The monitoring in Jordan is necessary in order to interrupt program execution – which execution occurs because of the use of Jordan's emulator -- and is of very specific parameters:

> the detector component 33 determines whether one or more of the indicators are triggered. Detector component 33 detects installation of a new structured exception handler (e.g., writing to address fs:[0]) [step 22] followed by forcing of the corresponding exception (step 23). Detector component 33 also detects use of the SIDTR instruction and/or modification of the IDT (step 24) followed by forcing of the corresponding interrupt (step 25).
>
> (Jordan, at paragraph 28, page 3.)

Accordingly, Applicant respectfully requests the withdrawal of the rejection to claim 1 and allowance of the claim.

Claims 2-4 depend from claim 1, and Applicant submits the arguments above with regard to independent claim 1 also apply to dependant claims 2-4. Accordingly, Applicant respectfully requests the withdrawal of the rejection to those claims.

Claim 6 has an interpretation element. Jordan has no such element and in fact has no references to interpretation. Moreover, Applicant has found no reference to an interpreted table in Jordan. Accordingly, Applicant respectfully requests the withdrawal of the rejection to claim 6 and allowance of the claim.

Claims 5, 7-12 stand as rejected under 35 U.S.C. 103(a) as being unpatentable over Jordan as applied to claim 1 and 4, and further in view of Shieh, et. al. (U.S. Patent No. 5,278,901) filed April 30, 1992 ("Shieh.") Applicant respectfully traverses the rejection.

Insofar as claim 5 depends from claim 1, Applicant submits the arguments above with regard to independent claim 1 also apply to dependant claim 5. Accordingly, Applicant respectfully requests the withdrawal of the rejection to that claim.

As to the rejections to claim 7, Applicant respectfully refers to the argument above concerning the differences between the emulator of Jordan and an interpreter. As was shown above, the Examiner's equating the two was misplaced. Therefore, Applicant submits, the emulator of Jordan and interpreter of claim 7 cannot be equated, and so the rejection should be withdrawn.

The Examiner also states that paragraph 38 of Jordan contains the remaining elements of claim 7. However, as was discussed above that paragraph has only a very specific monitoring function – one that cannot be seen as equivalent to claim 7's reporter and evaluator.

Finally, the Examiner has proposed combining Jordan with Shieh. However, there is no teaching, suggestion nor disclosure in either reference of such a combination. Moreover, it is submitted, simply lifting out the pattern analysis element of Sheih is impermissible hindsight reconstruction and would destroy the Sheih reference.

Accordingly, Applicant submits, the rejection to claim 7 cannot stand. Jordan does not show elements of the claim, and the proposed combination of Jordan and Sheih is unworkable. Applicant respectfully requests the withdrawal of the rejection to that claim.

Dependant claims 8-12 depend from claim 7 and share all the limitations of the claim. Thus, Applicant submits, the arguments above with regard to independent claim 7 also apply to dependant claims 8-12. Accordingly, Applicant respectfully requests the withdrawal of the rejection to those claims.

CONCLUSION

Therefore, for the reasons given above, Applicant submits the application is now in condition for allowance and Applicant respectfully requests early issuance of the Notice of Allowance.

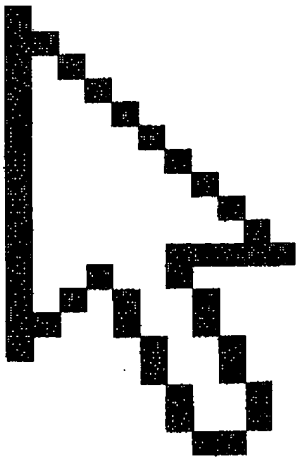
Respectfully submitted,

Joseph E. Chovanes
Registration No. 33,481
Suite 329
5 Great Valley Parkway
Malvern, PA 19355
(610) 648-3994

Microsoft

Microsoft Computer Dictionary

Fifth Edition



within a larger device or system. An embedded system is often built onto a single chip or board and is used to control or monitor the host device—usually with little or no human intervention and often in real time. *See also* microprocessor.

em dash *n.* A punctuation mark (—) used to indicate a break or interruption in a sentence. It is named for the em, a typographical unit of measure that in some fonts equals the width of a capital M. *Compare* en dash, hyphen.

EMF *n. See* electromotive force.

emitter *n.* In transistors, the region that serves as a source of charge carriers. *Compare* base (definition 3), collector.

emitter-coupled logic *n.* A circuit design in which the emitters of two transistors are connected to a resistor so that only one of the transistors switches at a time. The advantage of this design is very high switching speed. Its drawbacks are the high number of components required and susceptibility to noise. *Acronym:* ECL.

EMM *n. See* Expanded Memory Manager.

e-money or **emoney** *n.* Short for electronic **money**. A generic name for the exchange of money through the Internet. *Also called:* cybercash, digicash, digital cash, e-cash, e-currency.

emotag *n.* In an e-mail message or newsgroup article, a letter, word, or phrase that is encased in angle brackets and that, like an emoticon, indicates the attitude the writer takes toward what he or she has written. Often emotags have opening and closing tags, similar to HTML tags, that enclose a phrase or one or more sentences. For example: <joke>You didn't think there would really be a joke here, did you?</joke>. Some emotags consist of a single tag, such as <grin>. *See also* emoticon, HTML.

emoticon *n.* A string of text characters that, when viewed sideways, form a face expressing a particular emotion. An emoticon is often used in an e-mail message or newsgroup post as a comment on the text that precedes it. Common emoticons include :-) or :) (meaning "I'm smiling at the joke here"), ;-) ("I'm winking and grinning at the joke here"), :-(("I'm sad about this"), :-7 ("I'm speaking with tongue in cheek"), :D or :-D (big smile; "I'm overjoyed"), and :-O (either a yawn of boredom or a mouth open in amazement). *Compare* emotag.

EMS *n.* Acronym for Expanded Memory Specification. A technique for adding memory to PCs that allows for increasing memory beyond the Intel 80x86 microprocessor real-mode limit of 1 megabyte (MB). In earlier versions of microprocessors, EMS bypassed this memory board limit with a number of 16-kilobyte banks of RAM that could be accessed by software. In later versions of Intel microprocessors, including the 80386 and 80486 models, EMS is converted from extended memory by software memory managers, such as EMM386 in MS-DOS 5. Now EMS is used mainly for older MS-DOS applications because Windows and other applications running in protected mode on 80386 and higher microprocessors are free of the 1-MB limit. *Also called:* LIM EMS. *See also* expanded memory, protected mode. *Compare* conventional memory, extended memory.

em space *n.* A typographical unit of measure that is equal in width to the point size of a particular font. For many fonts, this is equal to the width of a capital M, from which the em space takes its name. *Compare* en space, fixed space, thin space.

emulate *vb.* For a hardware or software system to behave in the same manner as another hardware or software system. In a network, for example, microcomputers might emulate terminals in order to communicate with mainframes.

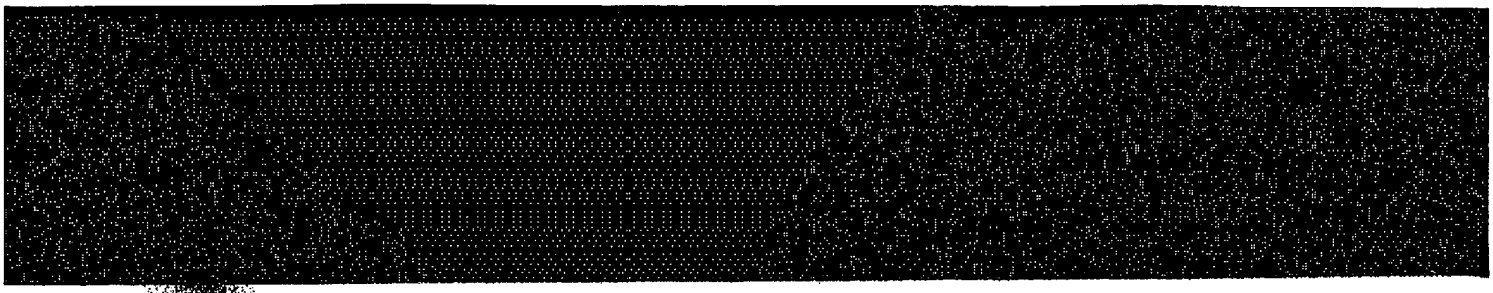
emulation *n.* The process of a computer, device, or program imitating the function of another computer, device, or program.

emulator *n.* Hardware or software designed to make one type of computer or component act as if it were another. By means of an emulator, a computer can run software written for another machine. In a network, microcomputers might emulate terminals in order to communicate with mainframes.

emulsion laser storage *n.* A method for recording data in film by selective heating with a laser beam.

enable *vb.* To activate or turn on. *Compare* disable.

encapsulate *vb.* **1.** To treat a collection of structured information as a whole without affecting or taking notice of its internal structure. In communications, a message or packet constructed according to one protocol, such as a TCP/IP packet, may be taken with its formatting data as an undifferentiated stream of bits that is then broken up and packaged according to a lower-level protocol (for example, as ATM packets) to be sent over a particular network; at the destination, the lower-level packets are assembled, re-creating the message as formatted for the encapsulated protocol. *See also* ATM (definition 1). **2.** In object-oriented



InterNIC *n.* Short for NSFnet (Internet) Network Information Center. The organization that has traditionally registered domain names and IP addresses as well as distributed information about the Internet. InterNIC was formed in 1993 as a consortium involving the U.S. National Science Foundation, AT&T, General Atomics, and Network Solutions, Inc. (Herndon, Va.). The latter partner administers InterNIC Registration Services, which assigns Internet names and addresses.

Interoperability *n.* Referring to components of computer systems that are able to function in different environments. For example, Microsoft's NT operating system is interoperable on Intel, DEC Alpha, and other CPUs. Another example is the SCSI standard for disk drives and other peripheral devices that allows them to interoperate with different operating systems. With software, interoperability occurs when programs are able to share data and resources. Microsoft Word, for example, is able to read files created by Microsoft Excel.

Interpolate *vb.* To estimate intermediate values between two known values in a sequence.

Interpret *vb.* **1.** To translate a statement or instruction into executable form and then execute it. **2.** To execute a program by translating one statement at a time into executable form and executing it before translating the next statement, rather than by translating the program completely into executable code (compiling it) before executing it separately. *See also* interpreter. *Compare* compile.

Interpreted language *n.* A language in which programs are translated into executable form and executed one statement at a time rather than being translated completely (compiled) before execution. Basic, LISP, and APL are generally interpreted languages, although Basic can also be compiled. *See also* compiler. *Compare* compiled language.

Interpreter *n.* A program that translates and then executes each statement in a program written in an interpreted language. *See also* compiler, interpreted language, language processor.

Interprocess communication *n.* The ability of one task or process to communicate with another in a multitasking operating system. Common methods include pipes, semaphores, shared memory, queues, signals, and mailboxes. *Acronym:* IPC.

Inter-record gap *n.* An unused space between data blocks stored on a disk or tape. Because the speed of disks and tapes fluctuates slightly during operation of the drives, a new data block may not occupy the exact space occupied by the old block it overwrites. The inter-record gap prevents the new block from overwriting part of adjacent blocks in such a case. *Acronym:* IRG. *Also called:* gap, interblock gap.

Interrogate *vb.* To query with the expectation of an immediate response. For example, a computer may interrogate an attached terminal to determine the terminal's status (readiness to transmit or receive).

Interrupt *n.* A signal from a device to a computer's processor requesting attention from the processor. When the processor receives an interrupt, it suspends its current operations, saves the status of its work, and transfers control to a special routine known as an interrupt handler, which contains the instructions for dealing with the particular situation that caused the interrupt. Interrupts can be generated by various hardware devices to request service or report problems, or by the processor itself in response to program errors or requests for operating-system services. Interrupts are the processor's way of communicating with the other elements that make up a computer system. A hierarchy of interrupt priorities determines which interrupt request will be handled first if more than one request is made. A program can temporarily disable some interrupts if it needs the full attention of the processor to complete a particular task. *See also* exception, external interrupt, hardware interrupt, internal interrupt, software interrupt.

Interrupt-driven processing *n.* Processing that takes place only when requested by means of an interrupt. After the required task has been completed, the CPU is free to perform other tasks until the next interrupt occurs. Interrupt-driven processing is usually employed for responding to events such as a key pressed by the user or a floppy disk drive that has become ready to transfer data. *See also* interrupt. *Compare* autopolling.

Interrupt handler *n.* A special routine that is executed when a specific interrupt occurs. Interrupts from different causes have different handlers to carry out the corresponding tasks, such as updating the system clock or reading the keyboard. A table stored in low memory contains pointers, sometimes called vectors, that direct the processor to the various interrupt handlers. Programmers can create interrupt handlers to replace or supplement existing handlers,